DOI: 10.1561/9781638285076.ch5

Chapter 5

Decentralized Identity Management

By Angel Palomares Perez and Laura Esbri Vidal

The ERATOSTHENES project aims to revolutionize identity management in IoT systems by leveraging Self-Sovereign Identity (SSI) principles, which allow IoT devices to maintain control over their digital identities throughout their lifecycle. Traditional centralized identity management systems often face scalability and security issues when applied to IoT environments, which contain a vast number of interconnected devices. By contrast, SSI offers a decentralized approach that not only enhances security and privacy but also simplifies identity management processes.

The Ledger uSelf SSI solution, a core innovation of the ERATOSTHENES project, plays a pivotal role in this decentralized framework. This solution integrates key components such as the PUF client, VDR-fabric, Advanced Data Protection (ADP) module, and the Identity Recovery Mechanism to provide a holistic identity management system. Through the creation of Decentralized Identifiers (DIDs) and Verifiable Credentials (VC), IoT devices are empowered with unique, cryptographically secure identities that can be used for authentication and authorization. These identities are self-managed, allowing devices to independently prove their identity while maintaining privacy, reducing the risks of tracking, profiling, or identity theft.

The importance of SSI in IoT environments lies in its ability to address the specific challenges that come with managing billions of devices, each with different hardware and software requirements. Ledger uSelf has been designed to

accommodate these constraints, offering flexibility through environmental configurations that support the deployment across various devices. Furthermore, the system's integration with advanced cryptographic techniques, such as privacy-enhancing Attribute-Based Credentials (p-ABC), ensures minimal data disclosure and enhances security through zero-knowledge proofs.

By applying SSI principles, the ERATOSTHENES project not only ensures secure device onboarding, identity verification, and authorization processes but also builds a zero-trust framework that aligns with modern privacy regulations. The use of disposable identities further bolsters privacy by creating unique identifiers for specific interactions, minimizing the risk of tracking.

In conclusion, the ERATOSTHENES project demonstrates the potential of SSI in transforming identity management for IoT devices, offering a scalable, privacy-centric, and secure solution that addresses the challenges of the increasingly interconnected digital landscape. The Ledger uSelf solution exemplifies the project's commitment to advancing secure, decentralized identity management in the IoT domain.

5.1 Introduction

Decentralized Identity represents a paradigm shift in how individuals, machines and IoT devices manage and control their digital identities. Traditional identity systems are typically centralized or federated, where a central authority, such as a government or large tech company, is responsible for verifying and controlling an individual's identity. This centralized approach is vulnerable to data breaches, privacy violations, and the concentration of power in single entities, leading to concerns about trust, security, and privacy.

Since IoT environments are typically distributed and highly dynamic, decentralized identity management has emerged as a transformative solution for addressing the challenges posed by centralized identity systems. Key principles such as self-sovereignty, verifiable credentials, trustless networks, and data minimization enable IoT devices to authenticate and interact securely, without over-exposing sensitive data. In contrast to traditional systems, decentralized identity reduces the risks of data breaches, enhances user control, and scales better with the exponential growth of IoT devices, making it a vital innovation in securing the future of IoT networks.

One of the primary objectives of the ERATOSTHENES project is to implement a Decentralized Identity Management system based on the Self-Sovereign Identity (SSI) paradigm throughout the entire lifecycle of IoT devices.

5.2 Overview of Decentralized Identity Frameworks and Standards

The Self-Sovereign Identity (SSI) community is still in an early stage of definition and implementation, and as a result, many of its main standards are continuously evolving, adapting, and being refined. Despite these changes, there is consensus within the community regarding the core building blocks needed to design and implement solutions following the SSI paradigm. These standards are essential for ensuring interoperability and security in decentralized identity systems. This chapter outlines the main findings related to these standards and how they will be applied in the solution developed for the ERATOSTHENES project.

The building blocks for SSI can be classified into tree main groups: standards that define the format of identity information, standards that define how identity information is transmitted between different actors, stakeholders and credential standard formats.

5.2.1 Identity Information Format Standard

In this category, two key standards from the **World Wide Web Consortium (W3C)**[1] play a crucial role:

- **W3C – Decentralized Identifiers (DID)**[2]: This standard defines a method for generating unique identifiers in a decentralized environment. DIDs are Uniform Resource Identifiers (URIs) that link a DID subject (such as a person, organization, or IoT device) with a DID Document. This DID Document contains cryptographic information, including public keys, which enable trusted interactions with the subject. The cryptographic proofs facilitate services like verification and authentication, essential for securing digital identities.

  The DID specification also introduces DID methods, which define how the DIDs are created, resolved, updated, and deactivated across various decentralized registries. The ERATOSTHENES project will implement two DID methods:

  - did:web[3]: This DID method, defined by W3C, uses standard web infrastructure to host DID Documents. It is particularly useful when

1. https://www.w3.org/standards/.
2. https://www.w3.org/TR/did-core/.
3. https://github.com/w3c-ccg/did-method-web.

decentralized registries are not needed, allowing DIDs to be hosted on trusted websites.

- did:erat: This is an ad-hoc DID method developed specifically for the ERATOSTHENES project. It stores DID Documents within Hyperledger Fabric, a permissioned blockchain framework developed for the project. This approach ensures that identities are managed securely within a controlled, permissioned environment.

A Decentralized Identifier (DID) follows a structured format, as illustrated in the image below. It is a simple text string consisting of three main components: the DID URI scheme identifier, the identifier for the specific DID method, and the method-specific unique identifier.

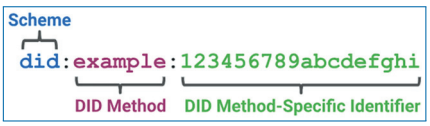

Figure 5.1. Example of a decentralized identifier (DID).

As mentioned before, the DID resolves to a DID Document that contains the DID and the associated cryptographic information such us public keys or verification methods, which are essential for authentication and verification processes. In the figure below shows an example of a DID Document that uses the "did:erat" method.


```
{
  "didDoc": {
    "@context": [
      "https://www.w3.org/ns/did/v1"
    ],
    "id": "did:erat:d56c97d236e93337dd3ec896d5185358595d57edd91d63da4fd2715eb0abd45d:9545a4009f8a8723402035f455879f6b0962167722d9f9e9d56fc
    "verificationMethod": [
      {
        "controller": "did:erat:d56c97d236e93337dd3ec896d5185358595d57edd91d63da4fd2715eb0abd45d:9545a4009f8a8723402035f455879f6b096216772
        "id": "did:erat:d56c97d236e93337dd3ec896d5185358595d57edd91d63da4fd2715eb0abd45d:9545a4009f8a8723402035f455879f6b0962167722d9f9e9d
        "publicKeyBase58": "4daZTzDC2VuWuYECFH549eJrGTVdZ4rv2BBChadAY8YERRXtdhNWzkWCpLX",
        "type": "Ed25519VerificationKey2018"
      },
      {
        "controller": "did:erat:d56c97d236e93337dd3ec896d5185358595d57edd91d63da4fd2715eb0abd45d:9545a4009f8a8723402035f455879f6b096216772
        "id": "did:erat:d56c97d236e93337dd3ec896d5185358595d57edd91d63da4fd2715eb0abd45d:9545a4009f8a8723402035f455879f6b0962167722d9f9e9d
        "publicKeyBase58": "3VZp2yNASMhXdLEeUwCoGQvSt3NvWRmbUm1WVFwzYNiQb99WJBxJy9JLwmBWCHm2nBiovFzaeuosaxJ5fvwB1GLhxuopGE4o32oKFc5nQjJKok
        "type": "X25519KeyAgreementKey2019"
      }
    ],
    "authentication": [
      "did:erat:d56c97d236e93337dd3ec896d5185358595d57edd91d63da4fd2715eb0abd45d:9545a4009f8a8723402035f455879f6b0962167722d9f9e9d56fc130e
    ],
    "assertionMethod": [
      "did:erat:d56c97d236e93337dd3ec896d5185358595d57edd91d63da4fd2715eb0abd45d:9545a4009f8a8723402035f455879f6b0962167722d9f9e9d56fc130e
    ],
    "keyAgreement": [
      "did:erat:d56c97d236e93337dd3ec896d5185358595d57edd91d63da4fd2715eb0abd45d:9545a4009f8a8723402035f455879f6b0962167722d9f9e9d56fc130e
    ],
    "created": "2023-10-05T08:41:08.104246721Z",
    "updated": "2023-10-05T08:41:08.104246721Z"
  }
}
```


Figure 5.2. Example of a DID Document, did:erat method.

- **W3C – Verifiable Credentials (VC)[4]:** VC offer a cryptographic mechanism to express and verify claims about an entity (such as an IoT device or individual) in a privacy-preserving and machine-verifiable manner. These credentials are essential for ensuring that sensitive data, like identity or device attributes, can be shared securely across decentralized networks. VCs are structured into three main roles:
  - *Holder*: The entity possessing one or more verifiable credentials and generating verifiable presentations from them. For instance, in an IoT context, the holder could be a device carrying a certificate of authenticity. Holders include students, employees, and customers.
  - *Issuer:* The entity responsible for creating a verifiable credential by asserting claims about a subject. Examples include corporations, governments, nonprofit organizations, trade associations and individuals issuing credentials that confirm device provenance or user identity.
  - *Verifier:* The entity responsible for receiving and validating verifiable credentials, often in the context of access control or authentication. Examples include services that validate the credentials of devices before allowing them onto a secure network that might include employers, security personnel, and websites.
  - *Verifiable Data Registry*: a system that mediates the creation, verification, and revocation of credentials. It maintains public keys, credential schemas, and revocation registries to ensure that credentials are valid and trustworthy.

The following figure shows the different roles in action and how they interact with each other.

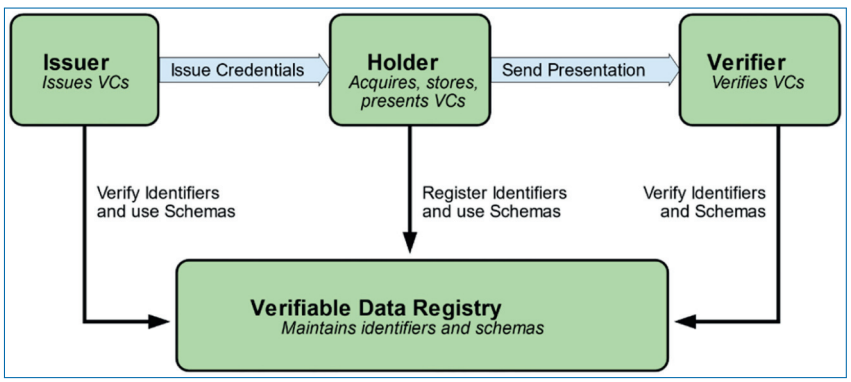

Figure 5.3. W3C Verifiable Credentials description.

4. https://www.w3.org/TR/vc-data-model/.

5.2.2 Communication Protocols for Decentralized Identity

In addition to defining the format of identity information, the Self-Sovereign Identity (SSI) community has proposed several standards to address how identity information that is transmitted between different roles and stakeholders. Two prominent standards in this area are:

- **DID Communication Messaging (DID Comm) v2.0**[5] is a communication protocol that facilitates secure and confidential messaging between stakeholders when interchanging identity information. It is designed to enable verifiable data exchange between SSI actors while maintaining privacy and security. This standard provides a robust mechanism for transmitting identity information across decentralized networks. DIDComm v2.0 defines protocols and workflows that specify how different SSI operations are executed, such as issuing credentials or presenting proof of a verifiable credential. These workflows enable the seamless exchange of identity-related information between holders, issuers, and verifiers. DIDComm also supports encryption, digital signatures, and message integrity checks, ensuring that identity information cannot be tampered with during transmission.

  Importantly, DIDComm v2.0 was specifically designed for Self-Sovereign Identity solutions, making it tailor-made for decentralized identity use cases. Major projects within the SSI community, such as Hyperledger Indy and Hyperledger Aries, utilize DIDComm to enable trusted communication between decentralized entities. By using DIDComm, decentralized identities can securely interact in a privacy-preserving way without needing intermediaries or central authorities. This communication framework allows for flexible and scalable identity management, which is essential for large-scale IoT systems, where billions of devices must interact securely.
- **Self-Issued OpenID Provider (SIOP) v2.0**[6] extends the widely adopted OpenID Connect protocol by introducing the concept of a Self-Issued OpenID Provider (Self-Issued OP). Unlike traditional OpenID Providers (OPs), which are controlled by third parties, a Self-Issued OP is fully controlled by the end-user. This means that individuals or entities can self-issue ID Tokens and present self-attested claims directly to Relying Parties (such as service providers), without needing a central authority to validate those claims. This approach aligns with the principles of SSI, as it empowers users to manage their own identities while interacting with services that rely on

5. https://identity.foundation/didcomm-messaging/spec/v2.0/.

6. https://openid.net/specs/openid-connect-self-issued-v2-1_0.html.

identity verification. SIOP v2.0 provides a way to bridge SSI solutions with existing identity management systems, making it possible to integrate modern, decentralized identities into traditional federated systems.

5.2.3 Credentials Format Standards

JSON Web Token (JWT)

As previously mentioned, the Self-Sovereign Identity (SSI) community is still in the early stages of defining and implementing its core concepts. Consequently, the primary standards are continuously evolving and adapting. During the ERATOSTHENES project, the W3C Verifiable Credential[7] standard was updated to support the JSON Web Token (JWT) format, which is a significant advancement.

JWT is an open standard that provides a compact and self-contained way of securely transmitting information between parties in the form of a JSON object. The data it carries can be verified and trusted because the tokens are cryptographically signed.

By incorporating support for JWT, the W3C Verifiable Credential standard simplifies the integration of verifiable credentials with existing Identity Management systems that already utilize the JWT format. This is particularly beneficial for systems based on OpenID Connect, making it easier to incorporate SSI principles without a complete overhaul of legacy infrastructure.

The following figures illustrate an example of how to apply the JWT format to verifiable credentials. The first Figure 5.4 displays a credential in JSON format, which includes the "*credentialSubject*" field that is used to hold the claims associated with the subject.

```
{
  "@context": [
    "https://www.w3.org/ns/credentials/v2",
    "https://www.w3.org/ns/credentials/examples/v2"
  ],
  "id": "http://university.example/credentials/3732",
  "type": ["VerifiableCredential", "ExampleDegreeCredential"],
  "issuer": "https://university.example/issuers/565049",
  "validFrom": "2010-01-01T00:00:00Z",
  "credentialSubject": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "degree": {
      "type": "ExampleBachelorDegree",
      "name": "Bachelor of Science and Arts"
    }
  }
}
```

Figure 5.4. Example of Verifiable Credential in JSON format.

7. https://www.w3.org/TR/vc-data-model-2.0.

Figure 5.5 illustrates the same verifiable credential as before, but this version includes the "*proof*" field, which contains the cryptographic data required to verify the integrity of the credential. The proof field ensures that the claims made within the credential can be trusted by allowing verifiers to check the credential's digital signature or other cryptographic elements. This feature has been present in earlier versions of the standard and remains a critical component for establishing trustworthiness in verifiable credentials.


```
{
  "@context": [
    "https://www.w3.org/ns/credentials/v2",
    "https://www.w3.org/ns/credentials/examples/v2",
    "https://w3id.org/security/suites/ed25519-2020/v1"
  ],
  "id": "http://university.example/credentials/3732",
  "type": [
    "VerifiableCredential",
    "ExampleDegreeCredential"
  ],
  "issuer": "https://university.example/issuers/565049",
  "validFrom": "2010-01-01T00:00:00Z",
  "credentialSubject": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "degree": {
      "type": "ExampleBachelorDegree",
      "name": "Bachelor of Science and Arts"
    }
  },
  "proof": {
    "type": "Ed25519Signature2020",
    "created": "2023-10-16T13:19:58Z",
    "verificationMethod": "https://university.example/issuers/565049#key-1"
,
    "proofPurpose": "assertionMethod",
    "proofValue": "zbEMsP6vpEKvwVvgv3uEe6u99hTUMm2MLhKPnZSsB9iy9KeXExUarCKr
95DrqCWpBqE8KDSSp3uJNXEk8Nhb2TLJ"
  }
}
```


Figure 5.5. Example of Verifiable Credential in JSON format with secured data integrity.

Figure 5.6 presents the same credential, this time shown in its decoded JSON Web Token (JWT) format. Readers familiar with tokens used in OpenID Connect will notice significant similarities between this format and the structure used in traditional Identity Management protocols. This resemblance allows for easier integration between modern Self-Sovereign Identity systems and legacy identity frameworks.

```
---------------- JWT header ----------------
{
  "alg": "ES256",
  "typ": "JWT"
}
---------------- JWT payload ----------------
// NOTE: The example below uses a valid VC-JWT serialization
//       that duplicates the iss, nbf, jti, and sub fields in the
//       Verifiable Credential (vc) field.
{
  "vc": {
    "@context": [
      "https://www.w3.org/ns/credentials/v2",
      "https://www.w3.org/ns/credentials/examples/v2"
    ],
    "id": "http://university.example/credentials/3732",
    "type": [
      "VerifiableCredential",
      "ExampleDegreeCredential"
    ],
    "issuer": "https://university.example/issuers/565049",
    "validFrom": "2010-01-01T00:00:00Z",
    "credentialSubject": {
      "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
      "degree": {
        "type": "ExampleBachelorDegree",
        "name": "Bachelor of Science and Arts"
      }
    }
  },
  "iss": "https://university.example/issuers/565049",
  "jti": "http://university.example/credentials/3732",
  "sub": "did:example:ebfeb1f712ebc6f1c276e12ec21"
}
```

Figure 5.6. Example of Verifiable Credential in decoded JSON Web Token format with secured data integrity.

Lastly, Figure 5.7 illustrates a verifiable credential encoded in JSON Web Token (JWT) format.

```
--------------- JWT ---------------
eyJhbGciOiJFUzI1NiIsInR5cCI6IkpXVCJ9.eyJ2YyI6eyJAY29udGV4dCI6WyJodHRwczovL3
d3dy53My5vcmcvbnMvY3JlZGVudGlhbHMvdjIiLCJodHRwczovL3d3dy53My5vcmcvbnMvY3JlZ
GVudGlhbHMvZXhhbXBsZXMvdjIiXSwiaWQiOiJodHRwOi8vdW5pdmVyc2l0eS5leGFtcGxlL2Ny
ZWRlbnRpYWxzLzM3MzIiLCJ0eXBlIjpbIlZlcmlmaWFibGVDcmVkZW50aWFsIiwiRXhhbXBsZUR
lZ3JlZUNyZWRlbnRpYWwiXSwiaXNzdWVyIjoiaHR0cHM6Ly91bml2ZXJzaXR5LmV4YW1wbGUvaX
NzdWVycy81NjUwNDkiLCJ2YWxpZEZyb20iOiIyMDEwLTAxLTAxVDAwOjAwOjAwWiIsImNyZWRlb
nRpYWxTdWJqZWN0Ijp7ImlkIjoiZGlkOmV4YW1wbGU6ZWJmZWIxZjcxMmViYzZmMWMyNzZlMTJl
YzIxIiwiZGVncmVlIjp7InR5cGUiOiJFeGFtcGxlQmFjaGVsb3JEZWdyZWUiLCJuYW1lIjoiQmF
jaGVsb3Igb2YgU2NpZW5jZSBhbmQgQXJ0cyJ9fX0sImlzcyI6Imh0dHBzOi8vdW5pdmVyc2l0eS
5leGFtcGxlL2lzc3VlcnMvNTY1MDQ5IiwianRpIjoiaHR0cDovL3VuaXZlcnNpdHkuZXhhbXBsZ
S9jcmVkZW50aWFscy8zNzMyIiwic3ViIjoiZGlkOmV4YW1wbGU6ZWJmZWIxZjcxMmViYzZmMWMy
NzZlMTJlYzIxIn0.QO_3_VhkpQC_nwgcSjXpG1PQHjilNtqPp0-3ylmM3exVGT9HbnoDXysGTWq
mCqXszgz9MFcT7Y6Vfedio7QuqQ
```

Figure 5.7. Example of Verifiable Credential in encoded JSON Web Token format with secured data integrity.

The addition of JSON Web Token (JWT) support to verifiable credentials simplifies their integration with legacy Identity Management systems by allowing credentials to be included in request headers, similar to OpenID and OAuth. The ERATOSTHENES consortium has adopted these changes to enhance their Self-Sovereign Identity solution for managing the entire lifecycle of IoT devices, from onboarding to decommissioning.

Selective Disclosure for JWT's (SD-JWT)

Selective Disclosure JWT (SD-JWT)[8] is an extension of the standard JWT that introduces the capability for selective disclosure of information. In SSI contexts, selective disclosure is a critical privacy feature because it allows users to share only specific pieces of their credentials with verifiers, rather than the full set of claims.

The process of issuing a Verifiable Credential with selectively hidden claims is outlined as follows. First, the Verifiable Credential (VC) is created. Then, the issuer designates specific claims as selectively disclosable by generating disclosures. These hidden claims are incorporated into the credential, producing an SD-JWT that conforms to a data model compatible with a Verifiable Credential containing concealed claims.

The process of issuing a Verifiable Credential with hidden claims is illustrated in Figure 5.8. First, a Verifiable Credential (VC) is created. After that, the issuer marks specific claims as selectively hidden, generating disclosures. These hidden claims are then embedded into the credential. The final result, known as SD-JWT, follows a data structure that ensures it is compatible with a Verifiable Credential containing concealed claims.

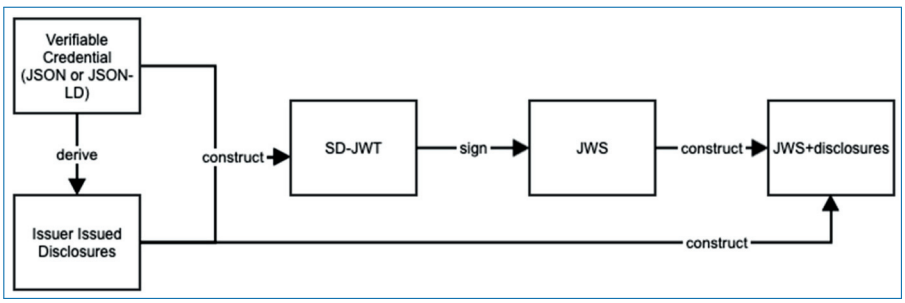

Figure 5.8. Issuing process following SD-JWT standard.

8. https://datatracker.ietf.org/doc/draft-ietf-oauth-selective-disclosure-jwt/.

The following section outlines the process of issuing a credential using real data. The first step involves collecting the necessary information and formatting it in accordance with the W3C Verifiable Credentials Data Model v1.1[9] standard. The figure below shows an example of a verifiable credential that is prepared for issuance.


```
{
  "@context": ["https://www.w3.org/ns/credentials/v2"],
  "id": "9bcc9aaa-3bdc-4414-9450-739c295c752c",
  "type": ["VerifiableCredential", "StudentID"],
  "issuer": "did:ebsi:zvHWX359A3CvfJnCYaAiAde",
  "validFrom": "2023-01-01T00:00:00Z",
  "validUntil": "2033-01-01T00:00:00Z",
  "credentialSubject": {
    "id": "did:key:z2dmzD81cgPx8Vki7JbuuMmFYrWPgYoytykUZ3eyqht1j9KbsDbVZXdb3jzCagESyY4EE2x7Y
    "familyName": "Carroll",
    "givenName": "Lewis",
    "birthDate": "1832-01-27",
    "student": true
  },
  "credentialSchema": {
    "id": "https://api-pilot.ebsi.eu/trusted-schemas-registry/v2/schemas/0x23039e6356ea6b703
    "type": "FullJsonSchemaValidator2021"
  }
}
```


Figure 5.9. Original Verifiable Credential before issuing process.

The next step involves calculating the disclosure values by applying the SHA-256 hashing method.


```
Content: '["2GLC42sKQveCfGfryNRN9w", "familyName", "Carroll"]'
calculated Disclosure: WyIyR0xDNDJzS1F2ZUNmR2ZyeU5STjl3IiwgImZhbWlseU5hbWUiLCAiQ2Fycm9sbCJd
calculated Digest: zSmImWHPJzQ7Rx8ZG0IYhUF1Ozj8f17wDKJGhxUkrdU


Content: '["eluV5Og3gSNII8EYnsxA_A", "givenName", "Lewis"]'
calculated Disclosure: WyJlbHVWNU9nM2dTTklJOEVZbnN4QV9BIiwgImdpdmVuTmFtZSIsICJMZXdpcyJd
calculated Digest: T4RnDm1clVLCav2Mrsel6sNMz8pqGCeMrrp__YrV_-w

Content: '["6Ij7tM-a5iVPGboS5tmvVA", "birthDate", "1832-01-27"]'
calculated Disclosure: WyI2SWo3dE0tYTVpVlBHYm9TNXRtdlZBIiwgImJpcnRoRGF0ZSIsICIxODMyLTAxLTI3Il0
calculated Digest: SFQTjr91IkPi6betQ0EYs5rdJ2TbMesJGftF6h7hjTA
```


Figure 5.10. Calculating disclose values.

9. https://www.w3.org/TR/vc-data-model/.

With the disclosed values, the Verifiable Credential is formed.


```
{
  "@context": ["https://www.w3.org/ns/credentials/v2"],
  "id": "9bcc9aaa-3bdc-4414-9450-739c295c752c",
  "type": ["VerifiableCredential", "StudentID"],
  "issuer": "did:ebsi:zvHWX359A3CvfJnCYaAiAde",
  "validFrom": "2023-01-01T00:00:00Z",
  "validUntil": "2033-01-01T00:00:00Z",
  "credentialSubject": {
    "id": "did:key:z2dmzD81cgPx8Vki7JbuuMmFYrWPgYoytykUZ3eyqht1j9KbsDbVZXdb3jzCagESyY4EE2x7Y
    "_sd": [
      "zSmImWHPJzQ7Rx8ZG0IYhUF1Ozj8f17wDKJGhxUkrdU",
      "T4RnDm1clVLCav2Mrsel6sNMz8pqGCeMrrp__YrV_-w",
      "SFQTjr91IkPi6betQ0EYs5rdJ2TbMesJGftF6h7hjTA"
    ],
    "student": true
  },
  "credentialSchema": {
    "id": "https://api-pilot.ebsi.eu/trusted-schemas-registry/v2/schemas/0x23039e6356ea6b703
    "type": "FullJsonSchemaValidator2021"
  },
  "_sd_alg": "sha-256"
}
```


Figure 5.11. Verifiable credential woth the disclosed calculated values.

For issuing the credential, it must be in JWT format. The process is illustrated in the image below.


```
eyJ0eXAiOiJKV1QiLCJhbGciOiJFUzI1NiIsImtpZCI6ImRpZDplYnNpOnp2SFdYMzU5QTNDdm-
ZKbkNZYUFpQWRlI0YwcjVPeXRfbGFodnZ6Nk1XbFlzM21jWU5LWmlpUWRVZnF2OHRzaEh-
OOXcifQ.eyJpc3MiOiJkaWQ6ZWJzaTp6dkhXWDM1OUEzQ3ZmSm5DWWFBaUFkZSIsInN1Yi-
I6ImRpZDprZXk6ejJkbXpEODFjZ1B4OFZraTdKYnV1TW1GWXJXUGdZb3l0eWtVWjNleXFodD-
FqOUtic0RiVlpYZGIzanpDYWdFU3lZNEVFMng3WWp4M2dOd2N0b0V1UkNLS0RyZE5QM0h-
QRnRHOFJUdkJpWVN0VDVnaEJlaEhpekgyRHk2eFF0VzNQZDJTZWNpekw5YjJqekRDTX-
I3S2E1Y1JBV1pGd3Zxd0F0d1RUN3hldDc2OXk5RVJoNiIsIm5iZiI6IjIwMjMtMDEtMDFUMDA6M-
DA6MDBaIiwiZXhwIjoiMjAzMy0wMS0wMVQwMDowMDowMFoiLCJqdGkiOiI5YmNjOWFhYS0z-
YmRjLTQ0MTQtOTQ1MC03MzljMjk1Yzc1MmMiLCJ2Yyl6eyJAY29udGV4dCI6WyJodHRwczov-
L3d3dy53My5vcmcvbnMvY3JlZGVudGlhbHMvdjIiXSwiaWQiOiI5YmNjOWFhYS0zYmRj-
LTQ0MTQtOTQ1MC03MzljMjk1Yzc1MmMiLCJ0eXBlIjpbIlZlcmlmaWFibGVBdHRlc3RhdGlvbiIsIl-
ZlcmlmaWFibGVDcmVkZW50aWFsIiwiU3R1ZGVudEIEll0sImlzc3VlciI6ImRpZDplYnNpOnp2SFd-
YMzU5QTNDdmZKbkNZYUFpQWRlIiwidmFsaWRGcm9tIjoiMjAyMy0wMS0wMVQwMDowM-
DowMFoiLCJ2YWxpZFVudGlsIjoiMjAzMy0wMS0wMVQwMDowMDowMFoiLCJjcmVkZW50aW-
FsU3ViamVjdCI6eyJpZCI6ImRpZDprZXk6ejJkbXpEODFjZ1B4OFZraTdKYnV1TW1GWXJXUGd-
Zb3l0eWtVWjNleXFodDFqOUtic0RiVlpYZGIzanpDYWdFU3lZNEVFMng3WWp4M2d-
Od2N0b0V1UkNLS0RyZE5QM0hQRnRHOFJUdkJpWVN0VDVnaEJlaEhpekgyRHk2eFF0VzN-
QZDJTZWNpekw5YjJqekRDTXI3S2E1Y1JBV1pGd3Zxd0F0d1RUN3hldDc2OXk5RVJoNiIsIl9z-
ZCI6WyJ6U21JbVdlUEp6UTdSeDhaRzBJWWhVRjFPemo4ZjE3d0RLSkdoeFVrcmRVIiwiVDRSb-
kRtMWNsVkxDYXYyTXJzZWw2c05NejhwcUdDZU1ycnBfX1lyVl8tdyIsIlNGUVRqcjkxSWtQaTZi-
ZXRRMEVZczVyZEoyVGJNZXNKR2Z0RjZoN2hqVEEiXSwic3R1ZGVudCI6dHJ1ZX0sImNy-
ZWRlbnRpYWxTY2hlbWEiOnsiaWQiOiJodHRwczovL2FwaS1waWxvdC5lYnNpLmV1L3Ryd-
XN0ZWQtc2NoZW1hcy1yZWdpc3RyeS92Mi9zY2hlbWFzLzB4MjMwMzllNjM1NmVhNml3MDNj-
ZTY3MmU3Y2ZhYzBiNDI3NjViMTUwZjYzZGY3OGUyYmQxOGFlNzg1Nzg3ZjZhMilsInR5cGUiOi-
JGdWxsSnNvblNjaGVtYVZhbGlkYXRvcjIwMjEifSwiX3NkX2FsZyI6InNoYS0yNTYifX0.wC1jmeb-
WUSVQYO18DluQhYUbTXDpvWuBSQv3apiiP3OdHlyPMRmFjX7jWPshjaWQkasuL_DBJ-
lQufgDzT8z7Q~WyIyR0xDNDJzS1F2ZUNmR2ZyeU5STjl3IiwgImZhbWIseU5hbWUiLCAi-
Q2Fycm9sbCJd~WyJlbHVWNU9nM2dTTklJOEVZbnN4QV9BIiwgImdpdmVuTmFtZSIs-
ICJMZXdpcyJd~WyI2SWo3dE0tYTVpVlBHYm9TNXRtdlZBIiwgImJpcnRoRGF0ZSIsICIxODMy-
LTAxLTI3Il0
```


Figure 5.12. Issued verifiable credential following SD-JWT standard.

Figure 5.12 shows an example that includes both the signed SD-JWT and the disclosure information, which are separated by a "~" character.

The following Figure 5.13 provides a detailed view of the format used.

```
<SD-JWT>~<Disclosure 1>~<Disclosure 2>~...~<Disclosure N>
```

Figure 5.13. Combined format for issuance.

SD-JWT-based Verifiable Credentials (SD-JWT VC)

This standard builds upon the SD-JWT approach by incorporating specific updates from the latest W3C Verifiable Credentials Data Model 2.0, which introduces several changes from previous versions. Among these updates is the addition of new parameters, such as the "vct" parameter, which defines the type of content in a Verifiable Credential (VC). The "vct" parameter dictates rules regarding which claims can or must be included in the Unsecured Payload of the SD-JWT VC,[10] and whether these claims can be selectively disclosed. While the standard does not specify preset "vct" values, it expects individual ecosystems to define them, including the corresponding semantics and rules for issuing and validating credentials.

The "vct" parameter is represented by a unique URI, such as "https://credentials.example.com/identity_credential". For instance, this credential type specifies that certain claims, like given_name, family_name, birthdate, and address, are mandatory in the Unsecured Payload. Additionally, other claims such as email, phone_number, and private claims like is_over_18 or is_over_21 may be included. The standard also allows selective disclosure of these claims when needed.

5.3 Ledger uSelf: Decentralized Identity Solution for IoT Devices

5.3.1 Architectural Positioning within the ERATOSTHENES Framework

The diagram in Figure 5.14 illustrates the ERATOSTHENES architecture, highlighting the main components and services developed by the project. The lower section of the diagram (in purple) represents the elements embedded in IoT devices, while the upper section shows the servers that host the various ERATOSTHENES framework services.

Focusing specifically on the components responsible for the project's Self-Sovereign Identity (SSI) solution, the architecture includes two key elements: the SSI Management service on the server side and the SSI Agent within the IoT

10. https://datatracker.ietf.org/doc/draft-ietf-oauth-sd-jwt-vc/

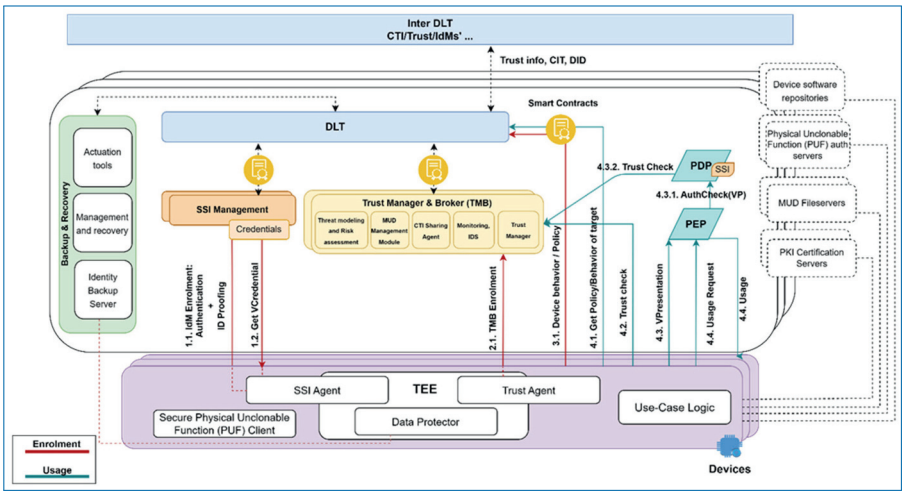

Figure 5.14. ERATOSTHENES general architecture diagram.

devices. The SSI Management service, known as the Ledger uSelf IdM component, operates on the server, while the SSI Agent, referred to as the Ledger uSelf IoT component, is deployed directly on the IoT devices. Although not explicitly shown in the diagram, the Context-aware component has been fully integrated within the Ledger uSelf IoT, ensuring seamless functionality across the SSI solution.

5.3.2 Integration of SSI in IoT Devices: Components Overview

The Ledger uSelf solution plays a pivotal role within the IoT device architecture, integrating several essential components that enhance both security and functionality:

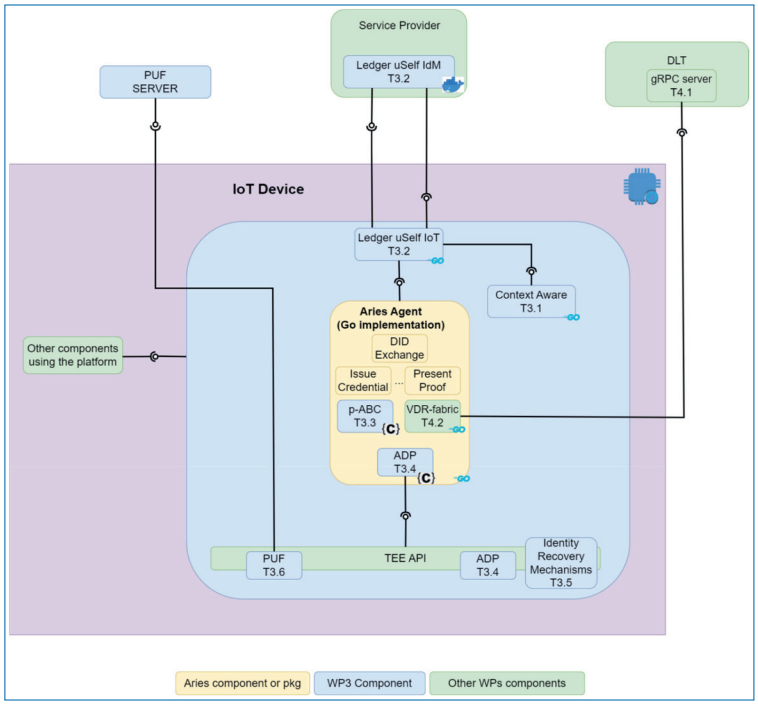

Figure 5.15. IoT components final integration for SSI solution.

- **Context-Aware Component:** This module enables the collection of crucial hardware and software information necessary to create a verifiable credential, which includes a unique device footprint for enhanced identification and authentication.
- **PUF Client and PUF Library:** These components utilize Physical Unclonable Functions (PUF), a form of hardware-based cryptography, to generate the public DID (Decentralized Identifier) associated with the device. This ensures robust device identification.
- **p-ABC Module:** This module provides advanced cryptographic capabilities that enable the generation of verifiable presentations and zero-knowledge proofs. This ensures that only necessary data is disclosed, protecting the privacy of the device's credentials.
- **Verifiable Data Registry Fabric (VDR-Fabric):** This module is responsible for generating public DIDs by leveraging a Hyperledger Fabric network. It stores DID documents, playing a crucial role in establishing the public DID for the device during the onboarding process.
- **Advanced Data Protection (ADP) Module:** The ADP module is designed to securely store identity information within the device. It also acts as a bridge to the Trusted Execution Environment (TEE), ensuring that sensitive identity data is protected.

- **Identity Recovery Mechanism:** This feature ensures that even in cases of device loss or replacement, the continuity and security of IoT data management are maintained, allowing for smooth identity recovery.

5.3.3 Key features of Ledger uSelf

5.3.3.1 Ledger uSelf Context Aware

The Context Aware component is specifically designed to assist in identifying IoT devices. While individual hardware and software characteristics alone may not be enough to distinctly identify a device, their combined use can form a unique profile. This combination of attributes creates a signature that helps in the recognition of each IoT device.

The key goal of this component is to generate a unique identifier that can be used in Authentication and Authorization services. In alignment with the Self-Sovereign Identity (SSI) model used by the ERATOSTHENES project, the unique footprint of the device is transformed into a verifiable credential. This verifiable credential is issued, signed, and secured by a Trusted Entity during the device's onboarding process. Once issued, this credential can be used within any SSI-compliant authentication framework.

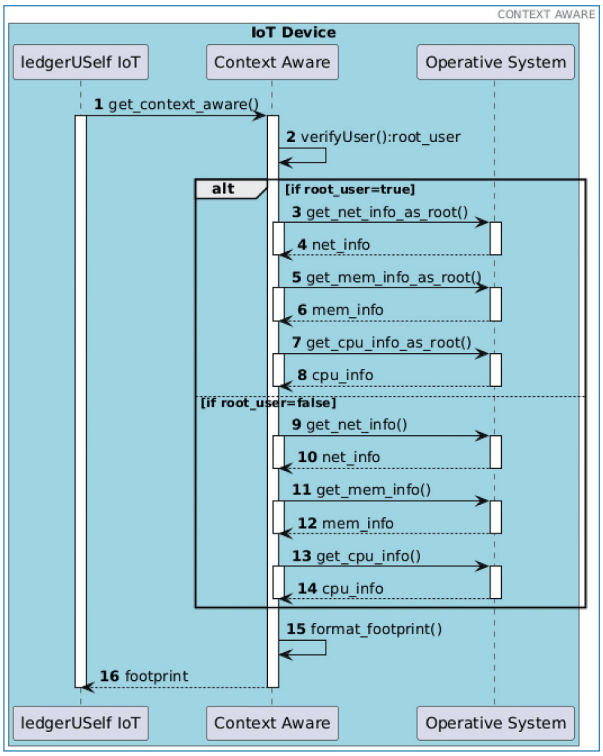

Figure 5.16. Context Aware sequence diagram.

Figure 5.16 illustrates the step-by-step process of gathering unique information from an IoT device, with access to this data dependent on the user's permission level. Superuser or root permissions allow for broader access to device information, while users with lesser permissions have more restricted access. This feature enables the service to generate distinct device profiles for each case. In the ERATOSTHENES project, it is expected that the process will typically run with superuser permissions. However, the component is also capable of generating a device footprint even when root access is not available, ensuring a flexible approach that strengthens the security of the overall process.

5.3.3.2 Ledger uSelf IoT Device

The primary purpose of the Ledger uSelf IoT component is to provide identity management for IoT devices within the ERATOSTHENES project, adhering to the principles of Self-Sovereign Identity (SSI). To meet the project's specific needs, this component has been designed as a single executable, integrating all necessary functions. This design choice simplifies the installation, management, and execution of the identity solution for IoT devices.

The final architecture of the SSI solution for the ERATOSTHENES project has been optimized to meet the project's unique requirements. Built on the Hyperledger Aries framework, the Ledger uSelf component acts as the core of the system, integrating with other essential project modules. Enhancements have been made to boost performance, add new features, and ensure smoother integrations. The solution is initially deployed in a containerized environment, which not only enhances operational efficiency but also facilitates better performance and scalability as new features are introduced.

The key features of the Ledger uSelf solution will be outlined in the following sections.

5.3.3.2.1 Create Public DID

The process of creating a Public Decentralized Identifier (DID) involves generating a unique identifier for the IoT device and developing a corresponding DID Document that adheres to the W3C DID specifications. This step is crucial for the onboarding process, as it establishes the device as a recognized participant within the system. The Identity Management component of the project plays a significant role in this functionality.

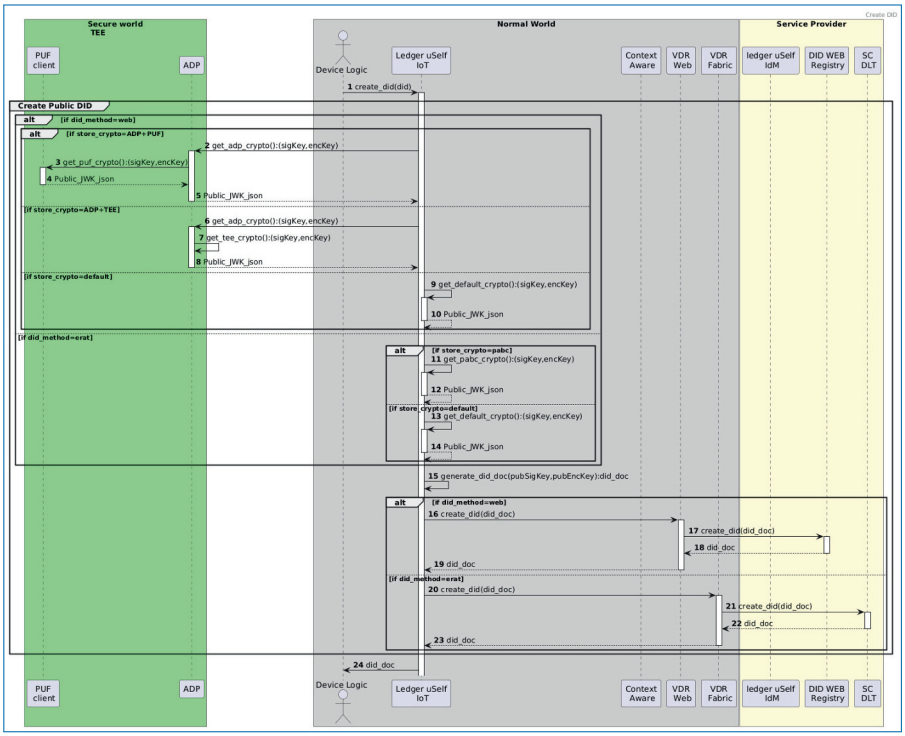

Figure 5.17. Create DID sequence diagram.

To accommodate the diverse needs of the ERATOSTHENES project, the system offers two methods for publishing the public DID:

- **did:web** This method follows established DID standards.
- **did:erat** A custom method specifically developed for the ERATOSTHENES project, this approach enables the DID to be published on a permissioned blockchain using Hyperledger Fabric. This ensures secure storage of decentralized identifiers while leveraging blockchain technology. The system incorporates the vdr-erat component, which includes a gRPC server and a Verifiable Data Registry (VDR), as illustrated previously in Figure 5.15.

The generation of cryptographic materials within the Ledger uSelf IoT is dynamic and involves three main sources:

1. PUF Client Library: This source produces physically unclonable cryptographic keys.
2. dP-ABC Module: This module offers cryptographic functionalities for generating derived verifiable presentations.
3. Hyperledger Aries: The framework manages its own cryptographic material generation.

5.3.3.2.2 Establish Connection

This method facilitates connections with other entities using the DID Exchange Protocol 1.0 and the Out-Of-Band Protocol (OOB) 1.1, as specified by Aries. Establishing a connection is essential for implementing any of the DID Communication standards.

To initiate a connection, each participant must generate a unique DID peer, followed by the exchange of these DIDs. This exchange enables secure communication between participants, utilizing the cryptographic methods linked to their respective DIDs.

From the device's perspective, the connection setup process is straightforward. The agent can simply invoke the connection functionality via the server-side URL, streamlining the establishment of the connection (see Figure 5.18 below).

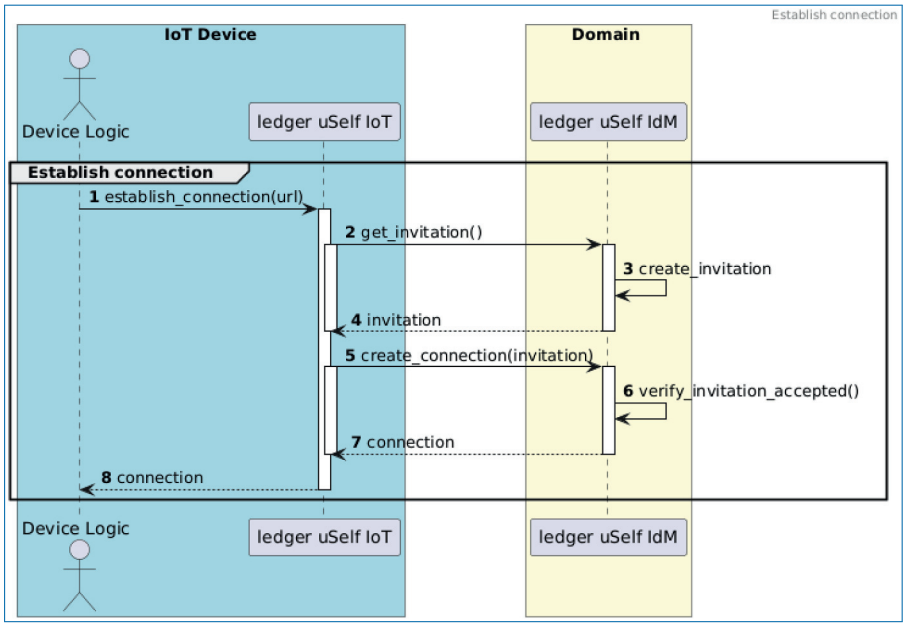

Figure 5.18. Establish Connection sequence diagram.

5.3.3.2.3 Device Onboarding

The management of Self-Sovereign Identity (SSI) is vital for the enrollment of IoT devices, as it involves issuing verifiable credentials that authenticate and authorize device identity attributes. Enrolling devices in the ERATOSTHENES framework formally registers them as members of its ecosystem, marking the second phase of the onboarding process, which begins with creating a Decentralized Identifier (DID).

The onboarding starts when the Ledger uSelf IoT component receives an onboarding request. Two primary tasks are performed: generating a public DID and registering it in the Identity Manager (IdM). The creation of a public DID includes generating a unique identifier for the device and constructing a corresponding DID Document that complies with the W3C DID standards. This process relies on cryptographic keys generated from three distinct sources, which are specified through environmental variables during installation.

The Ledger uSelf IoT component operates with a Hyperledger Aries agent, allowing flexibility to adapt to the specific requirements of the ERATOSTHENES project. After obtaining the necessary cryptographic keys, the system generates the DID Document using one of two methods: did:web or did:erat. Once the new DID is published, the device registers with the Ledger uSelf IdM agent and generates a verifiable credential that encapsulates its unique identity attributes, requesting the trusted entity to issue this credential (see Figure 5.19 below).

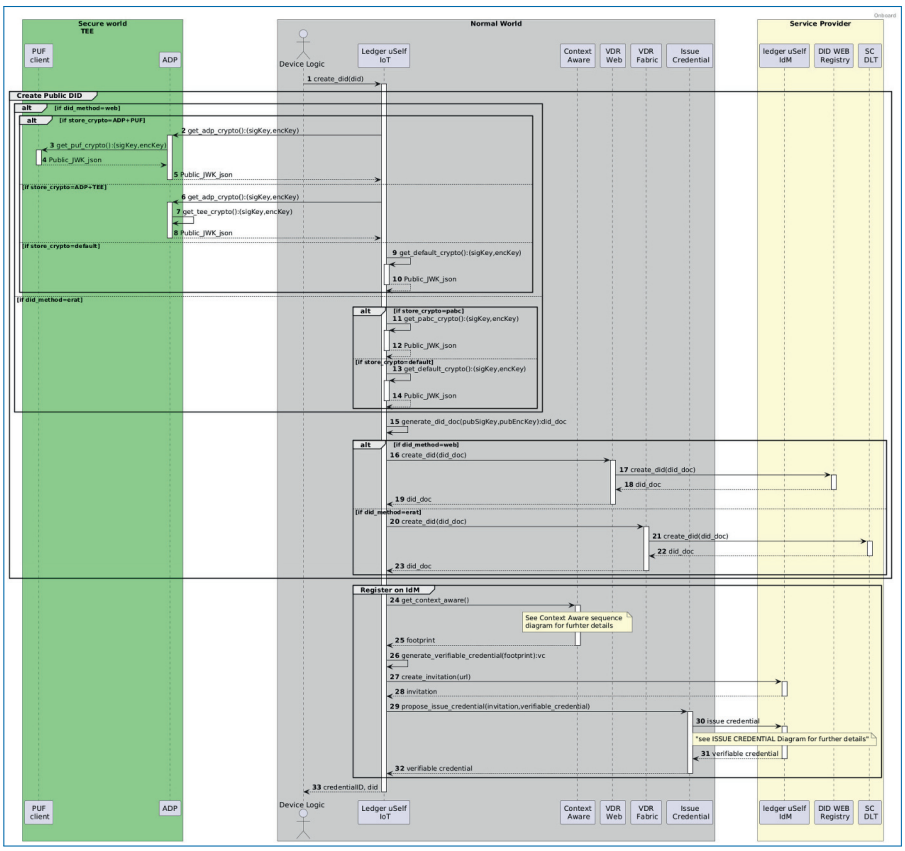

Figure 5.19. Onboarding sequence diagram.

5.3.3.2.4 Issue Credential

The implementation of this feature adheres to the guidelines established in the Issue Credential Protocol 2.0[11] from Hyperledger Aries and aligns with the DID Comm Messaging[12] standards set by the Identity Foundation. This protocol recognizes that communication between the Issuer and the Holder can be asynchronous, accounting for potential delays when the end user interacts with the system through a mobile or web wallet. However, adaptations have been made for situations requiring synchronous communication with IoT devices.

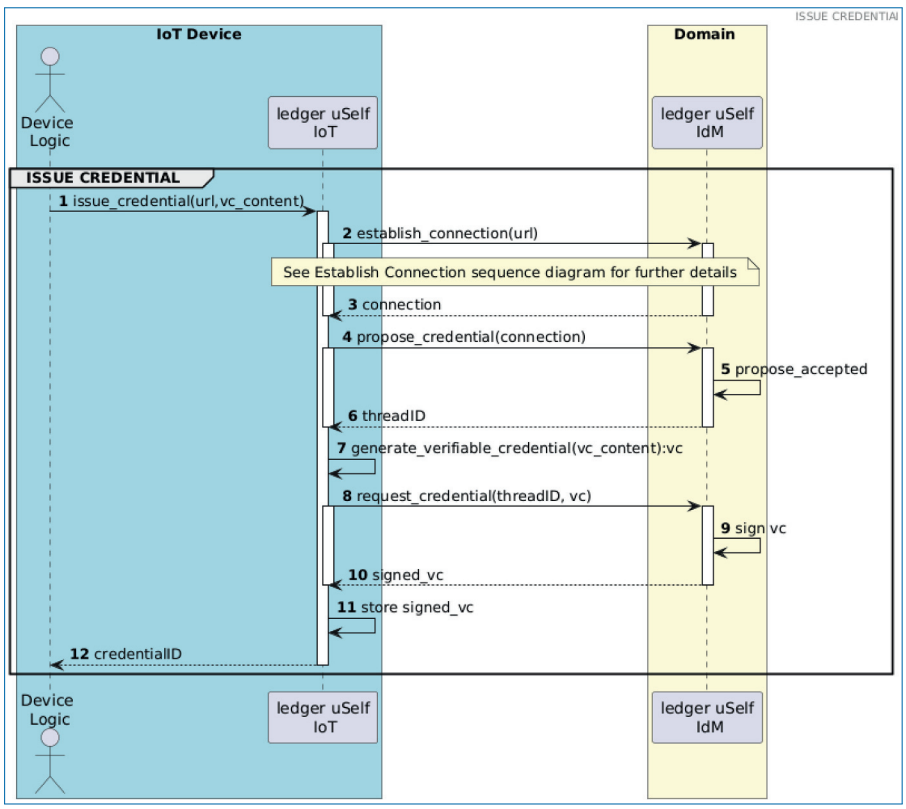

Figure 5.20. Issue Credential sequence diagram.

The process, illustrated in Figure 5.20, begins with the IoT device establishing a secure connection with the domain agent. Once the connection is in place, the Ledger uSelf IoT agent proposes to initiate the issuance process to the Ledger uSelf

11. https://github.com/hyperledger/aries-rfcs/tree/main/features/0453-issue-credential-v2.

12. https://identity.foundation/didcomm-messaging/spec/v2.0/.

IdM component, pending the issuer's agreement. This proposal includes a unique thread ID to track the process.

With the thread ID secured, the device then requests the Ledger uSelf IdM to issue the credential, providing the necessary verifiable credential content as a parameter. Upon receiving the signed verifiable credential from the issuer, the device stores it for future use.

5.3.3.2.5 Present Proof

The Present Proof feature enables IoT devices to prove ownership of a verifiable credential by presenting it to a verifier. This functionality is implemented in accordance with Hyperledger Aries' *Present Proof Protocol 2.0*[13] and complies with the DID Communication standards established by the Decentralized Identity Foundation (DIF).

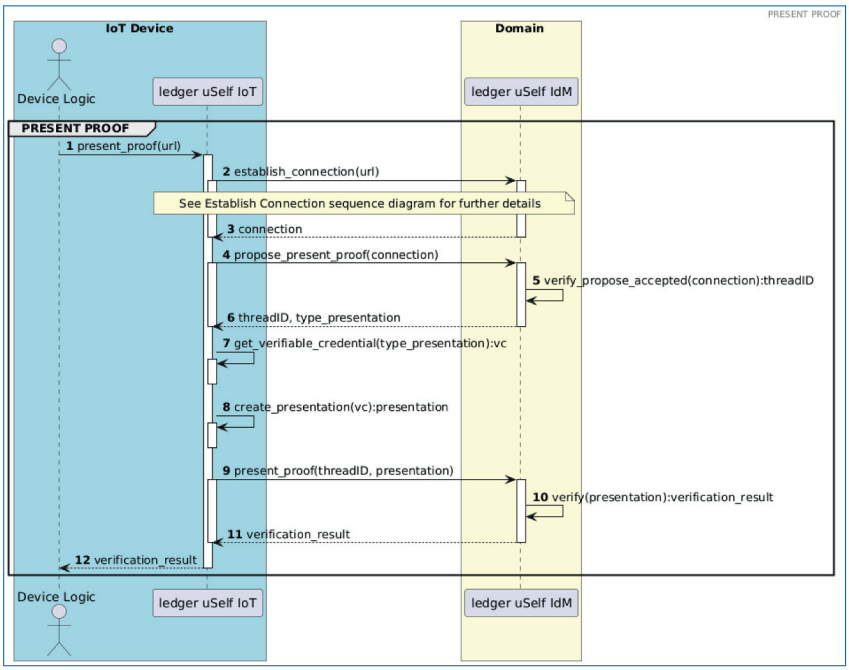

Figure 5.21. Present Proof sequence diagram.

The process typically begins with the device establishing a secure connection with the verifier. Once connected, the verifier provides a thread ID and defines the criteria for the proof. The device then searches its stored credentials to find one that matches the verifier's requirements. After identifying the appropriate credential, the

13. https://github.com/hyperledger/aries-rfcs/tree/main/features/0454-present-proof-v2.

device generates a verifiable presentation, demonstrating ownership of the credential. The presentation is then sent to the verifier for validation. If successful, the proof is accepted, allowing further steps in the process to proceed.

5.3.3.2.6 Get Verifiable Presentation

This feature allows IoT devices to create a verifiable presentation that proves their ownership and confirms their successful onboarding into the system. The process begins by retrieving the verifiable credential issued during onboarding. The device then uses this credential to generate a new verifiable presentation, which is re-signed for added security. Finally, the signed presentation is sent back to the device, confirming its authenticity and integration into the system.

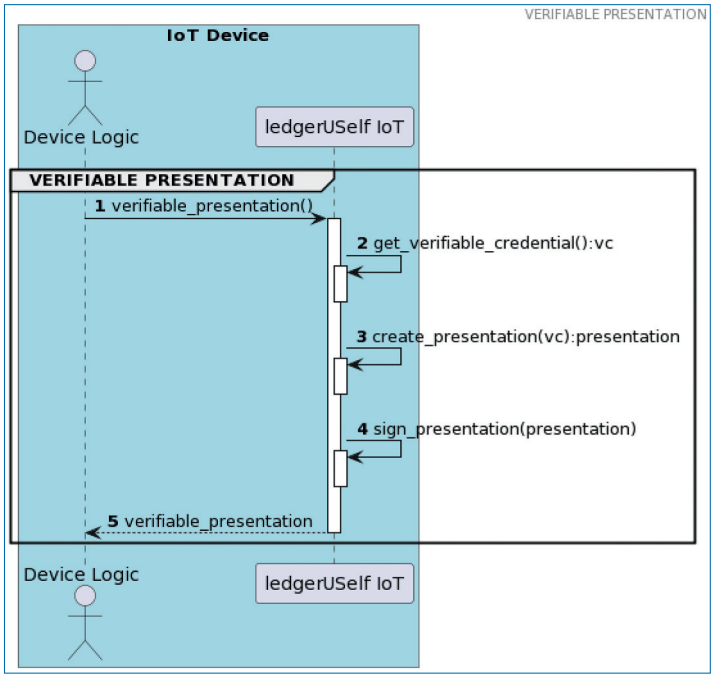

Figure 5.22. Get Verifiable Presentation sequence diagram.

5.3.3.2.7 Disposable ID

Disposable identities are temporary and limited-use identifiers, making them ideal for IoT devices in Self-Sovereign Identity (SSI) systems. Their short lifespan enhances privacy by preventing tracking and reducing the risk of identity theft, as a unique identifier is created for each interaction. This approach ensures that devices retain control over their data while complying with privacy standards.

In ERATOSTHENES, the disposable identity feature can be tailored to specific project needs through the configurable variable "USELF_EXPIRATION_TIME,"

which defines the identity's validity period in minutes. Once the IoT device is onboarded, its identity is created with a set expiration time. The "present proof" function is used to confirm if the identity remains valid. If expired, the system signals that the "verifiable presentation" is no longer valid. Additionally, the verifiable credential includes an "expirationDate" field, indicating when the credential will expire in a standard UTC format, ensuring clear timing for all parties involved.

5.4 Research and Scientific Innovation

In recent times, significant advancements have occurred in the realm of Self-Sovereign Identity (SSI), particularly within the European Union (EU). The EU has taken major steps towards establishing a unified framework for European Digital Identity. This initiative, aimed at benefiting all EU citizens, residents, and businesses, is grounded in decentralized identity management technologies. To support this, the EU has introduced a new regulation on electronic identification and trust services, known as eIDAS 2.0.[14] This regulation mandates that Member States provide digital wallets that link national digital identities with various personal attributes (such as driving licenses, diplomas, and bank accounts).

While the technical details of the European Digital Identity framework are still being refined, an important document, the *Architecture and Reference Framework (ARF)*,[15] outlines the key elements of the system. The ARF, in its latest version, defines the use of several core standards related to Self-Sovereign Identity. These include:

- **W3C Verifiable Credentials Data Model 1.1 (W3C VC-DM)**[16]: A framework for creating verifiable credentials that allow individuals to share trustworthy, tamper-evident digital credentials.
- **OpenID for Verifiable Credential Issuance (OpenID VCI)**[17]: A protocol for issuing verifiable credentials through OpenID Connect standards.
- **OpenID Connect for Verifiable Presentations (OpenID VP)**[18]: Facilitates the presentation of verifiable credentials using OpenID Connect.

14. https://digital-strategy.ec.europa.eu/en/policies/eidas-regulation.
15. https://github.com/eu-digital-identity-wallet/eudi-doc-architecture-and-reference-framework/blob/main/docs/arf.md.
16. https://www.w3.org/TR/vc-data-model/.
17. https://openid.net/specs/openid-4-verifiable-credential-issuance-1_0.html.
18. https://openid.net/specs/openid-4-verifiable-presentations-1_0.html.

- **Selective Disclosure for JWTs (SD-JWT)**[19]: Enables users to selectively disclose information from JSON Web Tokens (JWTs), ensuring greater privacy control.
- **SD-JWT-based Verifiable Credentials (SD-JWT VC)**[20]: Extends SD-JWT for verifiable credentials, allowing users to selectively disclose specific claims from their identity information.

The Ledger uSelf solution for the ERATOSTHENES project adheres to all relevant standards and integrates cutting-edge advancements in identity management technology. Its implementation includes innovations such as p-ABC (privacy-Attribute-Based Credentials) cryptography.

The solution holistically addresses the challenges of IoT identity management, starting from the creation of a root of trust for device identities, followed by enrollment, identification, and ongoing participation in a zero-trust and privacy-preserving framework for authorization. It supports Self-Sovereign Identity (SSI) principles for direct authentication and enables delegation of the authorization process to a Policy Decision Point (PDP) and Policy Enforcement Point (PEP) infrastructure. Furthermore, Distributed Ledger Technologies (DLTs) are employed to securely manage data, such as policies and trust scores, while preserving privacy using SSI methodologies.

An important innovation is the integration of p-ABCs with the W3C Verifiable Credentials (VC) specification. This combination allows for minimal disclosure and unlinkability, offering better privacy protection compared to other implementations like JWT-SD. This breakthrough addresses one of the previous limitations of p-ABCs, which lacked interoperability with other identity solutions, enhancing both scalability and efficiency.

Ultimately, the Ledger uSelf solution provides a flexible application of technologies, balancing security, privacy, and computational costs to cater to the diverse nature of IoT scenarios.

5.5 Conclusions

The ERATOSTHENES project has successfully achieved one of its core objectives: developing a decentralized identity management system grounded in Self-Sovereign Identity (SSI) principles for IoT devices throughout their lifecycle. This

19. https://datatracker.ietf.org/doc/draft-ietf-oauth-selective-disclosure-jwt/.

20. https://datatracker.ietf.org/doc/draft-ietf-oauth-sd-jwt-vc/.

SSI approach represents a significant leap forward, offering a decentralized, user-controlled identity management framework that contrasts with traditional centralized systems. By empowering IoT devices with autonomy over their identities, the solution greatly simplifies the complexities of device management while ensuring greater security and privacy.

In terms of technological development, the project concentrated on the integration and deployment of key components within IoT devices. The Ledger uSelf solution, in particular, serves as the central identity management component, seamlessly incorporating other essential features such as the PUF client, p-ABC cryptography, VDR-fabric, Advanced Data Protection (ADP) module, and the Identity Recovery Mechanism. These components were developed across multiple work packages, ensuring the solution meets the security, scalability, and performance needs of IoT environments.

A significant achievement in this project was the successful adaptation of these components to the specific hardware and software constraints of IoT devices, ensuring compatibility and ease of deployment. The modularity of the Ledger uSelf framework not only supports the devices used within ERATOSTHENES but also demonstrates the potential for broader application across various IoT systems. This adaptability allows for straightforward installation, execution, and management on different types of devices, offering a flexible, scalable solution that can be ported to other IoT contexts beyond the project.

In conclusion, the ERATOSTHENES project has created an innovative decentralized identity management framework that simplifies IoT device management while providing strong privacy, security, and scalability benefits, positioning it as a pioneering solution for the future of IoT identity systems.